LISTING OF CLAIMS

Please amend claims 9 and 22, as follows:

1. (Original) A method for processing a program statement in a database query language, the program statement corresponding to a plurality of operators, wherein an operator tree can be identified based upon the plurality of operators, the operator tree comprising a parent operator node, the parent operator node possibly associated with one or more child operator nodes, the method comprising:

(a) identifying whether one or more child nodes exist;

(b) for each of the identified one or more child nodes, determining if the child node relates to an operator for which top-down processing can be performed;

(c) calling and executing the operators from (a) for the child nodes that are eligible for top-down processing;

(d) generating output results for a child node that is not eligible for top-down processing; and

(e) outputting the output results to a data stream.

2. (Original) The method of claim 1 further comprising:

determining whether the data stream already exists; and

creating the data stream if it does not exist.

3. (Original) The method of claim 1 in which the program statement is intended to create XML, wherein one or more XML tags are generated.

4. (Original) The method of claim 3 in which the program statement comprises a SQL/XML operator.

5. (Original) The method of claim 4 in which the SQL/XML operator is a XMLElement(), XMLAgg(), XMLConcat(), XMLForest(), XMLAttribute(), XMLComment(), or XMLPI() operator.

6. (Original) The method of claim 1 in which nodes corresponding to a concatenate operation or a CASE WHEN statement on top of SQL/XML operator are eligible for top-down processing.

7. (Original) The method of claim 1 in which the data stream is closed after the parent operator node has been fully evaluated.

8. (Original) The method of claim 1 in which a child operator node is identified which is not eligible for top-down processing.

9. (Currently Amended) The method of claim 8 in which the child operator node not eligible for top-down processing is evaluated using bottom-up processing.

10. (Original) The method of claim 8 in which both top-down and bottom-up processing are used to evaluate the program statement.

11. (Original) The method of claim 1 in which the data stream is built at an intended target location for the output results.

12. (Original) The method of claim 1 in which the data stream is a single data stream.

13. (Original) The method of claim 1 in which the data stream is built on a buffer, LOB, HTTP stream, segmented array, data socket, pipe, file, internet stream type, network stream type, or FTP stream.

14. (Original) The method of claim 1 in which an intermediate copy is not stored for the output results.

15. (Original) A method for processing a program statement, the program statement corresponding to a plurality of operators, wherein an operator tree can be identified based upon the plurality of operators, the operator tree comprising a parent operator node, the method comprising:

(a) determining whether the parent operator node is related to a first child operator node that is eligible for top-down processing; and

(b) evaluating the first child operator node with top-down processing if the child operator is eligible for top-down processing, wherein the output from the first child operator node is output to a data stream.

16. (Original) The method of claim 15 in which the program statement is intended to create XML, wherein one or more XML tags are generated.

17. (Original) The method of claim 16 in which the program statement comprises a SQL/XML operator.

18. (Original) The method of claim 17 in which the SQL/XML operator is a XMLElement(), XMLAgg(), XMLConcat(), XMLForest(), XMLAttribute(), XMLComment(), or XMLPI() operator.

19. (Original) The method of claim 15 in which nodes corresponding to a concatenate operation or a CASE WHEN statement over a SQL/XML operator are eligible for top-down processing.

20. (Original) The method of claim 15 in which an intermediate copy is not stored for the output from the first child operator node.

21. (Original) The method of claim 15 in which a second child operator node is identified which is not eligible for top-down processing.

22. (Currently Amended) The method of claim 21 in which the second child operator node not eligible for top-down processing is evaluated using bottom-up processing.

23. (Original) The method of claim 15 in which the data stream is built at an intended target location for the output from the first child operator node.

24. (Original) The method of claim 15 in which the data stream is a single data stream.

25. (Original) The method of claim 15 in which the data stream is built on a buffer, LOB, HTTP stream, segmented array, data socket, pipe, file, internet stream type, network stream type, or FTP stream.

26. (Original) A computer program product comprising a computer usable medium having executable code to execute a process for processing a program statement in a database query language, the program statement corresponding to a plurality of operators, wherein an operator tree can be identified based upon the plurality of operators, the operator tree comprising a parent operator node, the parent operator node possibly associated with one or more child operator nodes, the process comprising:

(a) identifying whether one or more child nodes exist;

(b) for each of the identified one or more child nodes, determining if the child node relates to an operator for which top-down processing can be performed;

(c) calling and executing the operators from (a) for the child nodes that are eligible for top-down processing;

(d) generating output results for a child node that is not eligible for top-down processing; and

(e) outputting the output results to a data stream.

27. (Original) A system for processing a program statement in a database query language, the program statement corresponding to a plurality of operators, wherein an operator tree can be identified based upon the plurality of operators, the operator tree comprising a parent operator node, the parent operator node possibly associated with one or more child operator nodes, the method comprising:

(a) means for identifying whether one or more child nodes exist;

(b) means for determining if the child node relates to an operator for which top-down processing can be performed for each of the identified one or more child nodes;

(c) means for calling and executing the operators from (a) for the child nodes that are eligible for top-down processing;

(d) means for generating output results for a child node that is not eligible for top-down processing; and

(e) means for outputting the output results to a data stream.

28. (Original) A computer program product comprising a computer usable medium having executable code to execute a process for processing a program statement, the program statement corresponding to a plurality of operators, wherein an operator tree can be identified based upon the plurality of operators, the operator tree comprising a parent operator node, the process comprising:

(a) determining whether the parent operator node is related to a first child operator node that is eligible for top-down processing; and

(b) evaluating the first child operator node with top-down processing if the child operator is eligible for top-down processing, wherein the output from the first child operator node is output to a data stream.

29. (Original) A system for processing a program statement, the program statement corresponding to a plurality of operators, wherein an operator tree can be identified based upon the plurality of operators, the operator tree comprising a parent operator node, the method comprising:

(a) means for determining whether the parent operator node is related to a first child operator node that is eligible for top-down processing; and

(b) means for evaluating the first child operator node with top-down processing if the child operator is eligible for top-down processing, wherein the output from the first child operator node is output to a data stream.